Mapping Phonemes to Acoustic Symbols and Codes Using Synchrony in Speech Modulation Vectors Estimated by the Travellingwave Filter Bank

Ashwin Rao
Travellingwave, Seattle, WA, USA
ashwin@travellingwave.com

Abstract

A hybrid vector representation for speech resonances is defined using the modulation model and the sum of sinusoids model. An adaptive filter bank, which enables localization of synchronous modulation tracking, robustly estimates temporal variations in these vectors, is then presented. The synchrony in modulations within and across resonance channels, is subsequently used to derive acoustic symbols and codes that map fundamental units of languages, phonemes. Such an acoustic-phonetic mapping has never been demonstrated before. It has potential applications in speech recognition and voice analytics.

Index Terms: acoustic symbols, acoustic codes, acoustic cues, the speech code, phoneme mapping, symbolic representation

1. Introduction

Big-data systems that are currently used in applications like speech recognition [1] lack human-like performance and efficiency - their accuracy is susceptible to model mismatch [2, 3], they fail to provide reliable feedback for error-correction [4, 1], and they are very expensive to develop and deploy [1].

To address these problems, research on finding new acoustic cues in speech, which better map phonemes, has been underway for over a century [5, 6, 7, 8]. Many of these approaches are motivated by the way humans recognize phonemes, followed by syllables, words, sentences, and meaning [9].

Major strides have been made by Fant [6], Liberman [10], Stevens [11], Allen [12], and others [13, 14]. Their speech analysis experiments primarily rely on acoustic features estimated using the spectrogram [15], the linear prediction spectrum [16], and auditory filter banks [17, 12].

Unfortunately, successful mapping of phonemes has not been possible yet, due to a) high variability of existing speech features across speakers, phoneme context, and noise [6, 14], and b) limitations of time-frequency analysis tools [15] to jointly model phoneme transitions and resonances [14].

This paper introduces three new concepts for acoustic-phonetic mapping. The first, called modulation vector, is a hybrid representation for speech resonances that combines features from sinusoidal models [18, 19] and a generalized modulation model [20, 21, 22]. The second is an adaptive filter bank that improves upon the Rao-Kumaresan algorithm [22]; which was modified by Mustafa and Bruce in [23]. Specifically, it addresses problems in [22, 23] associated with complex-valued signals, frequency tracking errors, and filter instability. Additionally, it employs resonance localization to track modulation vectors in speech; instead of tracking formants as in [24, 25, 26, 27, 23], or modulated components (envelope and positive instantaneous frequency) as in [21, 22], or individual frequency components as in [28, 29]. Finally, the third concept utilizes synchrony in modulation vectors, within and across sub-bands, for mapping phonemes to acoustic symbols and codes.

In the remaining sections, modulation vector is defined in section 2, the adaptive filter bank is described in section 3, phoneme mapping using synchrony is derived in section 4, and simulation results are presented in section 5; discussions and conclusion follow in sections 6 and 7 respectively.

2. Modulation Vector

In [21, 22], the k-th resonance in a speech signal, s[n], was expressed using the product of elementary signals [20] as

\[ s_k[n] = a_k e^{j2\pi f_k n} c_k[n] e^{j\beta_k[n] - j\hat{\beta}_k[n]} \]

(1)

where \( n \) is the time sample, \( a_k \), and \( f_k \) are the carrier amplitude, and frequency, \( \beta_k \) is the carrier frequency. \( \alpha_k \) and \( \beta_k \) are details in modulations around \( f_k \); hat stands for Hilbert transform [30]. Using Eqn.1, along with speech representations based on sum of sine waves [18, 19], a modulation vector is now defined as

\[ \hat{M}_k = (a_k, f_k, b_k, a_{f_k}, f_{f_k}, b_{f_k}, p_k)^T \]

(2)

where \( a_k, f_k, \) and \( b_k \) denote amplitude, frequency, and bandwidth parameters, which model \( s_k[n] \)’s spectral envelope; \( b_k \) is the bandwidth around \( f_k \); and \( p_k \) is \( s_k[n] \)’s pitch. The relationship between \( f_k \) and \( f_{f_k} \) may be understood from [21]; parameters modeling \( \alpha_k \) and \( \beta_k \) may be added using modulation spectrum [31, 32] and sub-space [33] related concepts.

Next, the elements of \( \hat{M}_k \) are transformed, so that their scales and regions of interest, match the ones used in auditory systems [4, 12, 34], as follows: for \( i = k \) and \( ck \), \( a_i \) is converted to decibel (dB) using \( 10 \log_{10} a_i \); \( f_i \) and \( b_i \) are mapped to the Mel scale using 2595 \( \log_{10}(1 + F_{0i}/700) \) [4]; \( a_i \) and \( b_i \) are capped at 200 dB and 400 Mel respectively; and \( p_{f,k} \) outside the pitch range of 80-300 Hz are excluded. These features finally form the modulation vector, \( \hat{M}_k \).

3. Travellingwave Filter Bank (TFB)

The TFB algorithm estimates and tracks \( M_{k,s} \), by drawing inspiration from the travellingwave on the basilar membrane in the human ear’s cochlea [34, 35]. Its ability to separate individual resonances, along with its hybrid representation, makes TFB superior to the spectrogram, for speech analysis.

Each channel of TFB (Fig.1) consists of a Dynamic Tracking Filter (DTF), whose feed-back loop includes a first-order Linear Prediction (LP) estimator [30] and a Non-linear Masker (NM). The DTF is preceded by an All Zero Filter (AZF), and coupled to a Modulation Feature Estimator (MFE). A non-linear encoder (NE) finally outputs \( M_k \) as per section 2. The basic
idea behind TFB is that each channel’s AZF-DTF combination tracks the localized resonance’s frequency, and the MFE estimates (and implicitly tracks) the modulations characterizing its associated sub-band.

### 3.1. Dynamic Tracking Filter

The DTF proposed is an advancement to the one in [22]. It is an adaptive single-resonance filter with a transfer function

\[
H_{Dk}(n, z) = \frac{1 - r_p}{1 - r_p e^{2 \pi f_k z}} \, ,
\]

where \( k \) is the channel number; \( n \) is the sample number; and \( r_p \) is the pole-radius. \( f_k \) is estimated by LP (using its pole-angle) based on the past \( L \) samples of DTF’s output. The improvements made are described next.

#### 3.1.1. Estimation of \( a_k[n] \), \( b_k[n] \), and Constant-Q Option

\( a_k[n] \) is set to be \( \sqrt{\sigma_{lp}^2} \), where \( \sigma_{lp}^2 \) is the LP error-variance, and \( b_k[n] \approx - \ln(r_p) f_k / \pi \) [22], where \( r_p \) is the LP pole-radius; \( f_k \) is the sampling frequency. Further, \( L \) can be made smaller, as \( k \) increases, to maintain a constant-Q [4] window. This will enable rapid and finer analysis at higher frequencies.

#### 3.1.2. Implementation for real-valued signals

The DTF is implemented using the difference function,

\[
s_k[n] = c_k[n] s_k[n - 1] + r_p^2 s_k[n - 2] + g_k[n] \tilde{z}_k[n] \, ,
\]

where \( \tilde{z}_k[n] \) is the input to the DTF, \( s_k[n] \) is the DTF’s output, and \( c_k[n] = 2 r_p \cos(2 \pi f_k[n]) \); the DTF’s gain at \( f_k[n] \) is set to unity by \( g_k[n] = (1 - r_p)^2 + r_p^2 - 2 r_p \cos(4 \pi f_k[n]) \). It avoids computation of the analytic signal [15], thereby overcoming Hilbert transform related problems [36].

#### 3.1.3. Non-linear Masker

The LP outputs from all channels are analyzed by NM (Fig.2) as follows: Get Masker (GM) sorts \( f_k[n] \)’s and gets the strongest unmasked channel, \( k_s \). Then Get Thresholds (GTs) compute \( \delta F_L = f_k[n] - f_{k_s-1}[n] \) and \( \delta F_U = f_{k_s+1}[n] - f_k[n] \) for the lower and upper channels respectively. By comparing \( \delta F_L \) and \( \delta F_U \) to a masking threshold, \( t_m \), masking indicators, \( M_L \), and \( M_U \), are computed next; to be set to 0 (if \( \delta F_{L/U} < t_m \)) or 1 (if \( \delta F_{L/U} \geq t_m \)). The Masking Filters (MFs) finally output

\[
f_{k_s-1}[n] = M_L f_{k_s-1}[n] + (1 - M_L) f_{k_s-1}[n - 1] \, ,
\]

\[
f_{k_s+1}[n] = M_U f_{k_s+1}[n] + (1 - M_U) f_{k_s+1}[n - 1] \, . (5)
\]

This process is repeated until there are no unmasked channels.

NM eliminates errors due to switching of frequency tracks. Also, it weights the frequency estimates at \( n-1 \) and \( n \), using the estimated masking thresholds. This ensures stability of the overall (TFB) filter bank, when the DTF frequencies come close to each other. It is different from the one in [23] that sets a limit to the maximum allowable frequency spacing between DTFs, which results in tracking errors.

#### 3.2. All Zero Filter

The transfer function for the \( k \)-th channel AZF is [22]

\[
H_{Ak}(n, z) = G_k[n] \prod_{l=1}^{K-1} \left( 1 - r_s e^{2 \pi f_k[n] z^{-1}} \right) \, ,
\]

where \( r_s \) is the radius of the AZF’s zero, \( f_k[n] \) is the frequency of its zero-location (obtained from other DTFs), and

\[
G_k[n] = \frac{1}{\prod_{l \neq k} \left( 1 - r_s e^{2 \pi f_k[n] - f_l[n]} \right)} \, . (7)
\]

normalizes the \( k \)-th DTF’s gain. The improvements made to AZF include stability (due to NM) and ability to handle real-valued signals. The latter results from AZF’s design using a cascade of \( K - 1 \) filters with the \( l \)-th cascade implemented as

\[
\tilde{s}_{kl}[n] = \frac{s_{klf}[n] - c_{klf}[n] s_{kl}[n - 1] + r_p^2 s_{kl}[n - 2]}{g_{kl}[n]} \, , (8)
\]

where \( s_{klf}[n] \) is the cascade’s input (\( s_{klf}[n] = s[n] \)), \( \tilde{s}_{klf}[n] \) is the output (\( \tilde{s}_{klf}[n] \) being the same as \( \tilde{s}_{kl}[n] \) for \( l = K-1 \)), \( c_{klf}[n] = 2 r_s \cos(2 \pi f_k[n]) \), and the normalizing gain factor is \( g_{kl}[n] = \sqrt{1 + r_p^2 - 2 r_p \cos(2 \pi (f_k[n] - f_l[n]))} \times \sqrt{1 + r_p^2 - 2 r_p \cos(2 \pi (f_k[n] - f_l[n]))} \), with \( g_{kl}[n] > 0 \).

#### 3.3. Modulation Feature Estimator

The \( k \)-th MFE derives a non-distorted sub-band spectrum, \( S_{kn}[f] \), by utilizing the spectrum, \( S_k[f] \), of the past \( L_p \) samples of \( s[n] \) (computed only once \( \forall k \), using the Fourier Transform [4]), along with left and right frequency band-edges,

\[
f_{kL}[n] = \arg \min_{f} S_{Ef}[f] \left\{ f_{kL}[n] < f < f_{kL+1}[n] \cap f_k[n] < f < f_k[n+1] \right\}
\]

\[
f_{kR}[n] = \arg \min_{f} S_{Ef}[f] \left\{ f_{kR}[n] < f < f_{kR+1}[n] \cap f_k[n] < f < f_k[n+1] \right\} \, . (9)
\]

respectively; where \( S_{Ef}[f] = S_k'[f] \)’s spectral envelope [4, 30]. Since \( f_k[n] \) is being tracked, this results in an implicit tracking of \( S_{kn}[f] \). If \( b_k[n] \) is then set to be \( f_{kR}[n] - f_{kL}[n] \), \( a_k[n] \) and

---

Figure 1: TFB’s resonance localized modulation tracking

Figure 2: NM’s masking when frequencies get close
\( f_{ck}[n] \) are subsequently estimated as \( a_{ck}[n] = \max |S_{kn}[f]| \) and \( f_{ck}[n] = \arg \max |S_{kn}[f]| \).

Pitch, \( p_k[n] \), is computed using \( S_{kn}[f] \) (past \( L_p \) samples), and a hybrid of known techniques [4]. Using \( p_k[n] \) and a full-band pitch estimate, \( P_k[n] \), a sub-band pitch indicator, \( P_k[n] \), is then defined as \( P_k[n] = 1 \) if \( p_k[n] = P_f[n] \); and 0 otherwise. As will be seen in section 4, the \( P_k[n] \) yield useful cues; not provided by existing methods that group non-resonance sub-band pitches to yield one global pitch [37].

**4. Modulation Synchonony**

Based on several observations of \( M_k[n] \), using mixed language, gender, and age speakers, it is clear that: the simultaneous evolution of \( M_k[n] \)'s elements (i.e. their synchrony), within and across channels, trace symbols that map phonemes. This “modulation synchrony” is now demonstrated using the fricative consonant, SH, having the vowel IY as its context.

For ease of explanation, and since traces of \( f_s \) and \( f_{ck} \) are similar for IY-SH-IY, let us restrict \( M_k \) to \( (a_k, f_s, b_k, p_k)^T \). Also, instead of using \( M_k = [M_k, M_k, M_k, M_k, M_k] \), etc., let us use \( a_k, f_s, b_k, \) and \( p_k \); that way \( a_4[t][2]-a_1[t] \) may be easily interpreted as amplitude difference between channels 4 and 1 at time \( t \).

\[
\begin{align*}
\text{Figure 3: Ideal symbol for SH (with IY on left and right)}
\end{align*}
\]

Fig. 3 displays the **acoustic symbol** that has been observed for IY-SH-IY. Let \( R^s=2:4:3 \) denote SH’s resonance region. The details of cues in Fig. 3 are then as follows.

For resonance: \( a_s \) exceeds \( a_1 \) by at least 19 dB; the maximum range of all \( a_s \) is at least 3 dB greater than the maximum range of \( a_2, a_3, \) and \( a_4 \); SH’s peak amplitude is greater than those of its adjoining IYs; \( f_2 \) and \( f_1 \) are above 1500 and 250 Mel respectively; only \( b_1 \) is above 125 Mel (\( b_2, b_3, b_4 \) are below 125 Mel); all \( P_k \) are absent for SH; and \( R^s \)’s duration is between 30 and 500 msecs. And for transition: durations (\( t_{1:2} \) and \( t_{3:4} \)) are between 10 and 100 msecs, and \( a_s \)'s rise and drops are greater than 5 dB. These (acoustic) cues may be expressed as

\[
\begin{align*}
\text{(10)} & \quad a_4[v] - a_1[v] \geq t^4_a (v \in R^s) \\
\text{(11)} & \quad w^s_{2:4}[v] - w_{2:4}[v] \geq t^3_a (v \in R^s, w^s_{2:4}[v] \neq 0) \\
\text{(12)} & \quad a_{max} - a_{max} \geq t^3_a, \quad a_{max} - a_{max} \geq t^3_a \\
\text{(13)} & \quad f_2[v] \geq t^1_i, \quad f_1[v] \geq t^2_i (v \in R^s) \\
\text{(14)} & \quad b_1[v] \geq t^3_i, \quad b_2[v] \leq t^2_i (j = 2, 3, 4; v \in R^s) \\
\text{(15)} & \quad P_i[v] = 0 (v \in R^s) \\
\text{(16)} & \quad t^4_s \leq (t_{1:2} - t_{2:4} \leq t^3_s \\
\text{(17)} & \quad a_4[t][2] - a_1[t] \geq t^4_i, \quad a_1[t][3] - a_1[t][4] \geq t^3_i \\
\end{align*}
\]

for SH, left IY, and right IY respectively; the thresholds \( t^4_i, t^1_i, t^2_i, \) and \( t^3_i \) \((j=1,2,\ldots)\) can be estimated using standard statistical [30] or deep learning [38] techniques. Earlier studies [14] that characterize SH by dominant high frequency energy, relative amplitude, and noise duration, have reported only cues that are similar to Eqns. 13, 12, and 16 respectively.

The set of cues in Eqns. 10:18 form the **acoustic code** for IY-SH-IY. Eqns. 10 and 11 that correspond to predominant features of the symbol in Fig. 3, which are necessary to characterize the phoneme, are called the main cues; and \( a_4[v]-a_1[v] \), \( w^s_{2:4}[v] \div w^s_{2:4}[v] \) are called main cue-features.

**5. Simulations**

\[
\begin{align*}
\text{Figure 4: Spectrogram (Left) and MFB Outputs (Right)}
\end{align*}
\]

\[
\begin{align*}
\text{Figure 5:} & \quad a_{s,5} \text{ (Left)} \quad \text{and} \quad f_{s,5}+b_1 \text{ (Right)}
\end{align*}
\]

\[
\begin{align*}
\text{Figure 6: All} & \quad b_5 \text{ (Left)} \quad \text{and} \quad \text{Sub-band Harmonic Tracks (Right)}
\end{align*}
\]

First, results of analyzing an utterance corresponding to IY-SH-IY, spoken by a male speaker, using a Motorola Z2 Force smart-phone, are presented. TFB parameters were \( r_s=0.9, r_m=0.99, L=120, \) and \( L_p=250; K=4 \) and \( f_s=8 \) KHz.

For this example, the spectrogram (widely used for acoustic-phonetic mapping [14]) is shown in Fig. 4 Left, and outputs of the Mel Filter Bank (MBF), which is the **de facto** standard for speech recognition feature extraction [4, 1], is shown in Fig. 4 Right. Apart from high frequency energy, found in many phonemes, they fail to yield other cues, specific to SH.

Other problems associated with them include: a) peak-picking the spectrogram or choosing the right MBF channels, to track resonances, is not trivial [24, 25, 26, 27], b) any chosen MBF filter’s center frequency, may not line up with the signal’s resonance, resulting in frequency estimation errors, and c) MBF’s triangular weighted averaging could bias estimates of cues based on energies - e.g., energy difference between the two

4760
“manually selected” channels (1 and 10), whose center frequencies are close to 1st and 4th formant locations, is only ≈ 20 dB, as opposed to the true value of ≈ 36 dB (computed manually).

In contrast, Fig.5 displays an entire set of cues that form an acoustic symbol, similar to Fig.3. Specifically, Fig.5 (Left) shows that at peak resonance (τf = 540 msecs), α4-α1 = 36 dB; and α2, α3, α4 are grouped together relative to their separation from α1 (w1 4/4 + w2 4/4 = 18 DB). Further, αmax = (αmax, αmax), and α’s transitions during rise (375:435 msecs) and drop (645:720 msecs) are steep (23 dB and 19 dB respectively). Fig.5 (Right) shows that in the resonance region, f2 > 1500 Mel, f1 > 250 Mel, and b1 > 100 Mel. Also, the values of f2, f3, and f4 are similar to those of their adjoining IYs.

Fig.6 (Left) displays all b1’s for this example. Notice that only b1 exhibits deviations during SH’s resonance. Further, observe that Fig.6 (Right) shows no harmonic lines (no b1’s) for any of SH’s resonances, whereas IY’s display mostly all P2, k.

Clearly, the example considered maps to the acoustic code of Eqns. 10:18, with thresholds: τ1 = 36, τ2 = 18, τ0 = 5, τ1 = 1500, τ2 = 250, τ0 = 100, τ1 = 400, τ2 = 70, and τ4 = 19.

In Fig. 7, the average (μ) of main cue-features, sampled at tR, as a function of signal-to-noise ratio (SNR = 10 log10 $\left(\frac{P_o}{P_n}\right)$), where $P_o$ is the speech power with silence excluded, and $\sigma^2$ is the noise power) is plotted; error-bars indicate ±σ. A comparison of μ − σ to thresholds, reveals that TFB is robust at SNRs ≥8 dB for white noise; at lower SNRs, at least one cue-feature’s μ − σ falls below threshold, and the symbol looses its predominant shape. For factory noise, due to its intermittent bursts, σ is relatively higher and TFB is robust only for SNRs ≥15 dB.

Thus, TFB has potential to extract symbols even in noise.

The effect of increasing TFB’s DTF bandwidth is shown in Fig. 8. A comparison of Fig. 8 and Fig. 5, shows that TFB is not very sensitive to the choice of rP. However, at very low values, α4b and f4b display relatively more fluctuations (due to energy leakage from other sub-bands), and the symbol gets distorted. On decreasing DTF bandwidths, TFB will fail to track f4b and not yield symbols; similar to fixed filter banks like MFB.

Finally, Table 1 lists the number of additions, multiplications, and total calculations, needed for TFB (with 4 channels) and MFB (with 10 channels); only the filtering algorithms for TFB and MFB are compared; for TFB, only DTF, NM, and AZF (shown in Fig. 1) are considered; and for MFB, the computation of FFT, Mel cepstrum, delta cepstrum, and delta-delta cepstrum [4], are ignored. As can be seen, MFB requires ≈ 25 times more computations every second, compared to TFB. Results of further analysis, more examples, and links to TFB source-code and data-sets (to enable reproducibility), are in [39].

6. Discussions and Future Work

The acoustic symbols and codes derived for all English language phonemes (documented in [39]) indicate that a) the latter may be mapped to unique context-dependent shapes (similar to Fig. 3) and machine-readable rules (similar to Eqns. 10:18), which the spectrogram and MFB fail to accomplish; and b) all acoustic cues reported in earlier studies [14] correlate well, but with only a sub-set of cues rendered by the symbols. However, experiments reveal that some of the code equations (e.g., Eqns. 12:14 and 18, for IY-SH-IY) are not always satisfied for speakers enunciating poorly [39]; reinforcing the challenge of variability in speech. Interestingly, the code structure resembles layers of linear transforms coupled with non-linearities, seen in deep learning neural networks [40, 38]. For instance, Eqn. 10 is a linear combination of αa1[t] and αa2[t], followed by a non-linearity (> τ4), where each αa is output of linear filters (convolutional AZF and recurrent DTF) in Fig. 1) that is non-linearly transformed (by NE in Fig. 1). These new insights may be used to estimate code thresholds, in a way that the resulting codes enable speech recognition systems to require lesser training data, and be more robust to training-testing model mismatch [2, 3].

Further, using the code equations, a confidence metric may be defined as $C_{acc} = \frac{#matching Cues}{#Total Cues} \times 100$. It may be extended to word levels, and subsequently used for enabling speech understanding [41] and multi-modal [42] systems, to generate feedback, such as: display choices when 77% ≤ Cacc < 100%, prompt “speak clearly” if Cacc < 55%, and prompt “please repeat” for Cacc < 55%. Even further, the codes may be used to perform advanced voice analytics [43]. For example, τ2 = 30 in Eqn. 10 indicates that SH was spoken loudly; τ2 = 5 in Eqn. 11 indicates that SH was enunciated clearly; and τ2 = 0 in Eqn. 12 implies that IY was louder than its following SH phoneme.

TFB’s design around just 4 channels, each using simple (1-pole DTF and 3-zeros AZF) filters, makes it highly attractive for low-cost hardware and software implementations. The fine-tuning of its 5 parameters may be viewed as time-frequency filtering [15] “matched” to the acoustic symbols of phonemes.

The time-alignments that form part of the acoustic symbols (e.g., t1, t2, t3, t4, in Fig. 3), are currently being manually computed. An algorithm to automatically estimate these is work in progress. It will also facilitate more detailed acoustic-phonetic analysis, across multiple languages, using a large data-set.

7. Conclusion

The synchrony in speech modulation vectors, estimated using TFB’s resonance localized over time, helps derive acoustic symbols and codes that map phonemes. The codes have potential to improve many aspects of current speech recognition systems.
8. References


